What is Claimed is:

1. A method for transparently writing to shared memory when debugging a multiple processor system, the method comprising the steps of:

creating a software memory map of the memory usage of a plurality of processors in the system to be debugged;

activating a first debug session associated with a first processor of the plurality of processors and at least a second debug session associated with a second processor of the plurality of processors;

detecting a write request to a shared memory location by the first debug session;

if the first processor associated with the first debug session has write access to the shared memory location

then

selecting the first processor to perform the write request;

else performing the following steps a-b:

a. searching the software memory map to determine if the second processor has write access to the shared memory location;

b. selecting the second processor to perform the write request; and

passing the write request initiated by the first debug session to the selected processor for execution.

2. The method of Claim 1 wherein the step of passing the write request comprises the steps of:

searching the software memory map for a second plurality of processors that have read access to the shared memory location;

broadcasting the write request to the second plurality of processors; and

performing cache coherency updates in response to the write request in each of the second plurality of processors.

3. The method of Claim 2 wherein the step of broadcasting the write request comprises indicating that the write request is intended for maintaining cache coherency as opposed to a normal write request.

4. The method of Claim 2 wherein the step of performing comprises using cache coherency capabilities, if any, of a processor in response to the write request intended for maintaining cache coherency.

5. The method of Claim 2 wherein:

the step of creating comprises denoting in the software memory map the shared memory locations that contain program instructions;

the step of passing the write request additionally comprises the step of determining that the shared memory location contains a program instruction; and

the cache is an instruction cache.

6. The method of Claim 1 wherein the step of passing the write request comprises:

determining that the write request is to a shared memory location at which a software breakpoint has been set;

searching the software memory map to find a third plurality of processors with read access to the shared memory location;

sending the third plurality of processors the new instruction for the shared memory location;

updating a software representation maintained for software breakpoints for each of the third plurality of processors to replace the old instruction with the new instruction; and

resetting the software breakpoint with the new instruction in the write request;

7. The method of Claim 6 wherein the step of resetting comprises:

clearing the software breakpoint at the shared memory location;

performing the write request; and

setting a software breakpoint at the shared memory location.

8. The method of Claim 2 wherein the steps of the method are performed iteratively to write to successive shared memory locations.

9. The method of Claim 1 further comprising the steps of reading the shared memory location by the first processor after the contents of the shared memory location have been changed by the write request.

10. A software development system, comprising:

a memory storage system holding a software development tool program;

a host computer connected to the memory storage system, the host computer operable to execute the software development tool program;

a test port for connecting to a hardware system, the hardware system being comprised of multiple processors with shared memory and operable to execute an application program; and

wherein the software development tool is operable to support debugging of the application program executing on the hardware system using a method for transparently writing to shared memory when debugging a multiple processor system, the method comprising the steps of:

creating a software memory map of the memory usage of a plurality of processors in the system to be debugged;

activating a first debug session associated with a first processor of the plurality of processors and at least a second debug session associated with a second processor of the plurality of processors;

detecting a write request to a shared memory location by the first debug session;

if the first processor associated with the first debug session has write access to the shared memory location

then

selecting the first processor to perform the write request;

else performing the following steps a-b:

a. searching the software memory map to determine if the second processor has write access to the shared memory location;

b. selecting the second processor to perform the write request; and

passing the write request initiated by the first debug session to the selected processor for execution.

11. The software development system of Claim 10 wherein the step of passing the write request comprises the steps of:

searching the software memory map for a second plurality of processors that have read access to the shared memory location;

broadcasting the write request to the second plurality of processors; and

performing cache coherency updates in response to the write request in each of the second plurality of processors.

12. A digital system, comprising:

multiple processors with common shared memory for executing an application program; and

wherein the application program was developed with a software development system using a method for transparently writing to shared memory when debugging a multiple processor system, the method comprising the steps of:

creating a software memory map of the memory usage of a plurality of processors in the system to be debugged;

activating a first debug session associated with a first processor of the plurality of processors and at least a second debug session associated with a second processor of the plurality of processors;

detecting a write request to a shared memory location by the first debug session;

if the first processor associated with the first debug session has write access to the shared memory location

then

selecting the first processor to perform the write request;

else performing the following steps a-b:

a. searching the software memory map to determine if the second processor has write access to the shared memory location;

b. selecting the second processor to perform the write request; and

passing the write request initiated by the first debug session to the selected processor for execution.

13. The digital system of Claim 12 wherein the step of passing the write request comprises the steps of:

searching the software memory map for a second plurality of processors that have read access to the shared memory location;

broadcasting the write request to the second plurality of processors; and

performing cache coherency updates in response to the write request in each of the second plurality of processors.